Министерство науки и высшего образования Российской Федерации Федеральное государственное бюджетное образовательное учреждение высшего образования

**Пермский национальный исследовательский политехнический университет**

Электротехнический факультет

Кафедра информационных технологий и автоматизированных систем

**ОТЧЕТ**

**Лабораторная работа №1**

Семестр: 3

Выполнил студент РИС-23-2б:

Серебряков Егор Константинович

(дата, подпись)

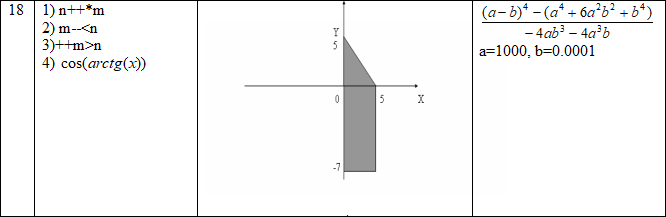
Проверила:

О. Л. Викентьева

(дата, подпись)

Пермь 2024

**Вариант №18**



**Задача №1**

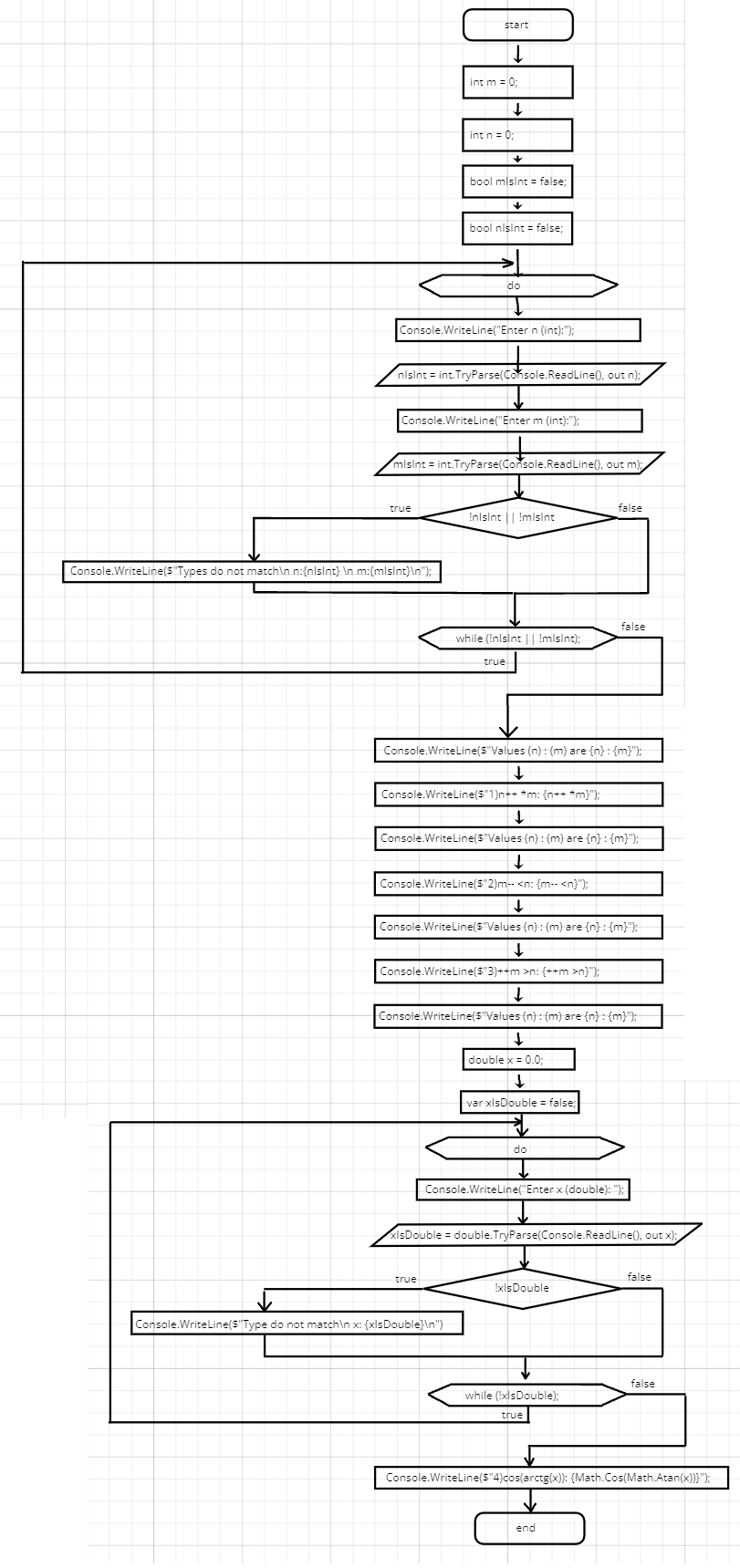
**Постановка задачи**

1. Для задачи 1 найти значения выражений. Если необходимо, то определить при каких исходных данных выражение не может быть вычислено и выдать сообщение об ошибке.
2. Составить систему тестов и вычислить полученное выражение для нескольких значений Х.

**Анализ задачи**

* Переменные n, m – целочисленного типа (int). В ходе инициализации n, m следует реализовать проверку типа данных.
* Переменная x – типа double, не имеет ограничений по области допустимых значений. Аналогично проводится проверка типа данных.
* После ввода корректных значений в переменные n,m и x выводятся соответствующие задаче результаты

**Блок схема**

****

**Код программы**

int n = 0;

int m = 0;

bool nIsInt = false;

bool mIsInt = false;

//Ввод и проверка на целочисленность переменных n,m

do

{

Console.WriteLine("Enter n (int):");

nIsInt = int.TryParse(Console.ReadLine(), out n);

Console.WriteLine("Enter m (int):");

mIsInt = int.TryParse(Console.ReadLine(), out m);

if (!nIsInt || !mIsInt) Console.WriteLine($"Types do not match\n n:{nIsInt} \n m:{mIsInt}\n");

}

while (!nIsInt || !mIsInt);

//Вывод результатов с переменными n,m

Console.WriteLine($"Values (n) : (m) are {n} : {m}");

Console.WriteLine($"1)n++ \*m: {n++ \*m}");

Console.WriteLine($"Values (n) : (m) are {n} : {m}");

Console.WriteLine($"2)m-- <n: {m-- <n}");

Console.WriteLine($"Values (n) : (m) are {n} : {m}");

Console.WriteLine($"3)++m >n: {++m >n}");

Console.WriteLine($"Values (n) : (m) are {n} : {m}");

double x = 0.0;

var xIsDouble = false;

//Ввод и проверка на тип double переменных x

do

{

Console.WriteLine("Enter x (double): ");

xIsDouble = double.TryParse(Console.ReadLine(), out x);

if (!xIsDouble) Console.WriteLine($"Type do not match\n x: {xIsDouble}\n");

}

while (!xIsDouble);

Console.WriteLine($"4)cos(arctg(x)): {(double)Math.Cos(Math.Atan(x))}");

**Результаты работы**

Рассмотрим результаты работы при следующих значениях:

* n = 10
* m = 15
* x = 1,235

Результаты:

1. В выражении «n++ \*m» сначала выполнится постфиксный инкремент, после выполнится умножение. Подставляя значения, выражение примет следующий вид: «10 \* 15».  
   Ответом будет целочисленное число 150.
2. В выражении «m-- <n» сначала выполнится постфиксный декремент, после выполнится проверка “<”. Подставляя значения, выражение примет следующий вид «15 < 10».  
   Ответом будет булевое значение false.
3. В выражении «++m >n» сначала выполнится префиксный инкремент, после выполнится проверка “>”. Подставляя значения, выражение примет следующий вид «16 > 10».  
   Ответом будет булевое значение true.
4. Выражение «cos(arctg(x))» имеет тригонометрические функции, следовательно в этом случае применим класс Math с методами Math.Cos и Math.Atan. Подставляя значения, выражение примет вид «cos(arctg(1,235))».  
   Результатом является значение типа double 0,629288738535062

**Задача №2**

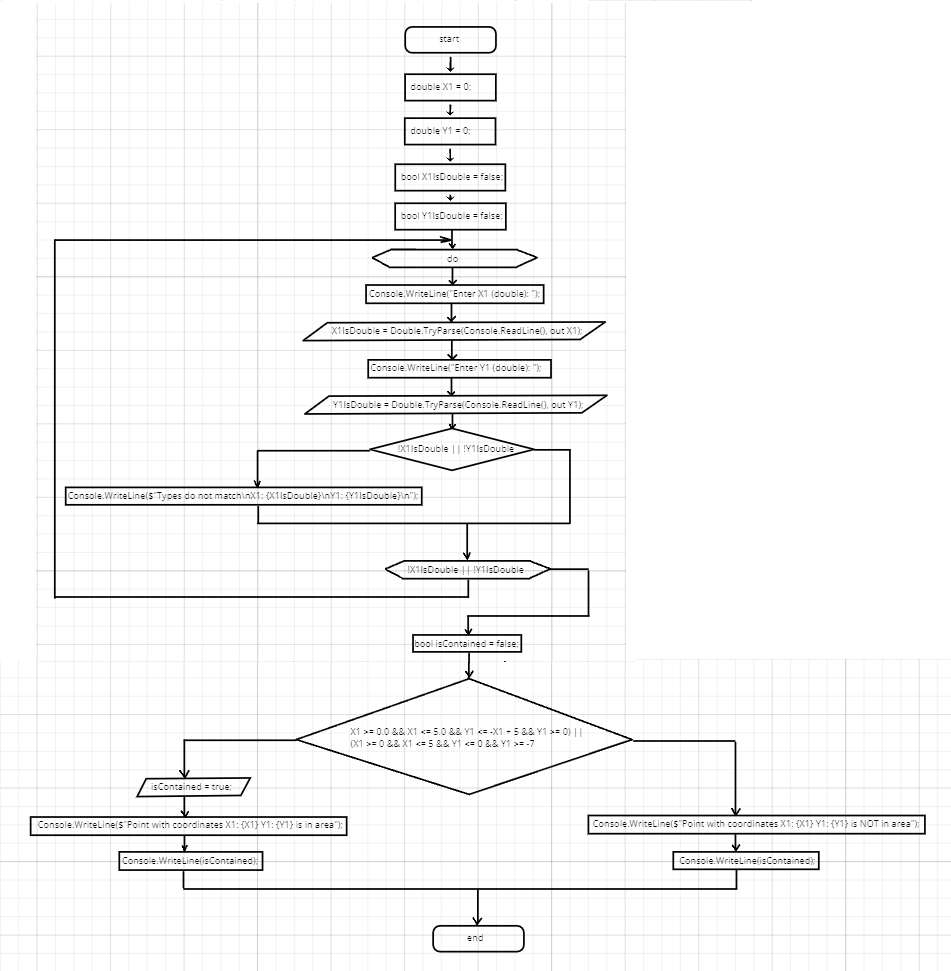
**Постановка задачи**

1. Для задачи 2 записать выражение, зависящее от координат точки X1 и Y1 и принимающее значение TRUE, если точка принадлежит заштрихованной области, и FALSE, если не принадлежит
2. Составить систему тестов и вычислить полученное выражение

**Анализ задачи**

* Переменные X1, Y1 – вещественного типа (double). В ходе инициализации n, m следует реализовать проверку типа данных.
* Для проверки вхождения координат точки в область следует задать условия. Условия определяются прямыми, при успешном прохождении всех проверок, выводится true, иначе false
* После ввода корректных значений в переменные X1, Y1 выводятся соответствующие задаче результаты

**Блок схема**

****

**Код программы**

double X1 = 0.0;

double Y1 = 0.0;

bool X1IsDouble = false;

bool Y1IsDouble = false;

//Ввод и проверка на тип double переменных X1,Y1

do

{

Console.WriteLine("Enter X1 (double): ");

X1IsDouble = Double.TryParse(Console.ReadLine(), out X1);

Console.WriteLine("Enter Y1 (double): ");

Y1IsDouble = Double.TryParse(Console.ReadLine(), out Y1);

if (!X1IsDouble || !Y1IsDouble) Console.WriteLine($"Types do not match\nX1: {X1IsDouble}\nY1: {Y1IsDouble}\n");

}

while (!X1IsDouble || !Y1IsDouble);

//Проверка на включение точки в обл.значений

bool isContained = false;

if((X1 >= 0.0 && X1 <= 5.0 && Y1 <= -X1 + 5 && Y1 >= 0) || (X1 >= 0 && X1 <= 5 && Y1 <= 0 && Y1 >= -7))

{

isContained = true;

Console.WriteLine($"Point with coordinates X1: {X1} Y1: {Y1} is in area");

Console.WriteLine(isContained);

}

else

{

Console.WriteLine($"Point with coordinates X1: {X1} Y1: {Y1} is NOT in area");

Console.WriteLine(isContained);

}

**Результаты работы**

Рассмотрим результаты работы при следующих значениях:

* X1 = 2,9999999999
* Y1 = 3,0

Результаты:

1. Значения переменных подставляются в следующие условия «X1 >= 0.0 && X1 <= 5.0 && Y1 <= -X1 + 5 && Y1 >= 0) || (X1 >= 0 && X1 <= 5 && Y1 <= 0 && Y1 >= -7»
2. «X1 >= 0.0» - выполняется, 2,9999999999 > 0.0
3. «X1 <= 5.0» - выполняется, 2,9999999999 < 5.0
4. «Y1 <= -X1 + 5» - не выполняется, 3.0 < 2,0000000001
5. Третье условие не выполняется, следовательно, результатом будет false

**Задача №3**

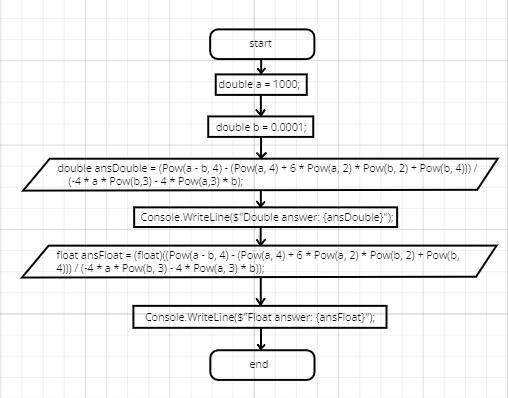
**Постановка задачи**

1. Для задачи 3 вычислить значение выражения, используя различные вещественные типы данных (float и double)
2. Результаты всех вычислений вывести на печать
3. Объяснить полученные результаты

**Анализ задачи**

* Переменные a и b следует задать в типе double, чтобы погрешность вычислений была минимальна
* Для вычисления значений в степенях следует использовать Math.Pow метод, который возвращает значение типа double
* Для сравнения результатов в одном из случаев вычисления результата следует написать явное преобразование в тип float, чтобы передаваемое значение соответствовало задаче

**Блок схема**



**Код программы**

double a = 1000;

double b = 0.0001;

//Получаем double результаты

double ansDouble = (Pow(a - b, 4) - (Pow(a, 4) + 6 \* Pow(a, 2) \* Pow(b, 2) + Pow(b, 4))) /

(-4 \* a \* Pow(b,3) - 4 \* Pow(a,3) \* b);

Console.WriteLine($"Double answer: {ansDouble}");

//Получаем float результаты

float ansFloat = (((float)Pow(a - b, 4) - (float)(Pow(a, 4) + (float)6 \* (float)Pow(a, 2) \* (float)Pow(b, 2) + (float)Pow(b, 4))) /

(-4 \* (float)a \* (float)Pow(b, 3) - 4 \* (float)Pow(a, 3) \* (float)b));

Console.WriteLine($"Float answer: {ansFloat}");

**Результаты работы**

Рассмотрим результаты работы при следующих значениях:

* a = 1000
* b = 0.0001

Результаты:

1. При подстановке в выражение типа double получается следующий ответ: 0,99999999999999
2. При подстановке в выражение типа float получается следующий ответ: 0,98304
3. Разница в точности объясняется тем, что для типа double выделено больше байт, следовательно, его мантисса больше, чем мантисса числа типа float (float – 4 byte, double – 8 byte)